Description

[A Command Multiplier for Built-In-Self-Test]

BACKGROUND OF INVENTION

[0001] Field of Invention

[0002] The present invention relates to the built-in-self-test of embedded memories, and more particularly, to methods of generating multiple sets of inputs from a single seed set of inputs enabling high-speed testing by a lower speed BIST circuit.

[0003] Background of Invention

[0004] Embedded memories require special test solutions due to the unique characteristics of memory circuits. Typically, integrated circuits (ICs) that contain embedded memories also contain a Built-In-Self-Test circuit (BIST) which is used to enable the testing of the memory by generating command, address, and data (CAD) patterns for exercising the memory. Using the BIST for this purpose enables the

use of an external logic tester to test the logic portion of the IC as well as the embedded memory.

[0005] Testing embedded memories at their highest operating speed (that is the shortest command cycle time) requires that the CAD patterns be generated at that high operating speed. However, as memory speeds increase, it becomes difficult for a BIST to calculate the CAD pattern in this short cycle time, particularly given the circuit area constraints placed on the BIST.

[0006] As embedded memory performance and complexity continues to increase, BIST testing at application speeds also becomes more important; hence, novel schemes that support increased BIST performance, while still maintaining BIST flexibility and minimizing design schedule and chip area are of great value. An architecture that allows for a single BIST that can test a diverse range of memory types and sizes at a number of different performance points would be advantageous.

SUMMARY OF INVENTION

[0007] One solution to this problem is to employ a "command multiplier" circuit (CMX), which is placed between the BIST and the memory. The CMX accepts CAD information from the BIST at a relatively low speed, generates additional

CAD information (i.e., "multiplies" the pattern data) and then delivers it to the memory at its maximum operating frequency. This has several clear advantages:

[0008] i)The BIST itself can operate at a relatively low speed.

[0009] ii)If a chip contains multiple embedded memories, then the CMX can be located locally to each memory, and the BIST can be shared among multiple memories, saving chip area.

[0010] iii)The BIST to CMX bus, which may be quite long, can run at the lower frequency, while only the much shorter CMX to memory bus will run at maximum frequency.

[0011] The general concept of command multiplication has been attempted as described in U.S. Pat. No. 6,647,524 entitled "Built-In-Self-Test Circuit for RAMBUS Direct RDRAM"(issued Nov. 11, 2003 to Huang and Kwai).

[0012] The present invention discloses apparatus and a method for generating CAD information which are flexible enough to allow in-depth characterization of data pattern related defects, while still allowing efficient circuit layout. At least one of the techniques enables multiple address generation for testing a memory with concurrent refresh or multiple ports.

[0013] A command multiplication scheme is described in which

CAD information provided by the low-speed BIST is used as a "seed" set for the generation of high-speed CAD patterns output to the memory. Specifically, the CMX described here takes seed CAD information from a BIST operating at a frequency, f_{BIST}, and generates n sets of CAD data. Each generated CAD set is then output in turn to a memory system at a frequency that is equal to f_{BIST} x n.

[0014] The CMX's CAD generation is accomplished by modifying the seed set based on the contents of previously loaded registers or previously set input signals. The modification can be anything from combining the seed set in a bit-wise fashion with the bits of a directive register through a logic unit (LU) to the addition of an offset using an arithmetic-logic unit (ALU). In the case of the latter, generating an address in a non-bit-wise manner is particularly important for simulating a worst-case noise condition in a multi-bank operation.

[0015] This invention is also directed to support multiple address generation from a common seed address using different address offsets for each address port. Generating multiple addresses is an important requirement when testing a memory with a concurrent refresh or multiple ports.

[0016] The inventive apparatus described may also be used to

generate complex CAD patterns including non-bit-wise address sequences and multiple address generation. This results in more complete test coverage for both single and multiple port memories.

BRIEF DESCRIPTION OF DRAWINGS

[0017] The invention will be better understood from the following detailed description with reference to the drawings in which:

[0018] Figure 1. A block diagram showing the interconnections of the BIST, CMX, and memory in accordance with the present invention.

[0019] Figure 2. A block diagram of a bit-wise command multi-plier in accordance with a first embodiment of the present invention;

[0020] Figure 3. A block diagram of a bit-wise command multi-plier in accordance with a second embodiment of the present invention.

[0021] Figure 4. A block diagram of a bit-wise command multi-plier in accordance with a third embodiment of the present invention.

[0022] Figure 5. A block diagram of a bit-wise command multi-plier in accordance with a fourth embodiment of the present invention.

[0023] Figure 6. An example of a timing diagram for CAD generation based on offsets where n equals 4; and

[0024] Figure 7. Tables illustrating CMX command and data op codes.

DETAILED DESCRIPTION

[0025] Detailed Description of Invention

[0026] Figure 1 shows the flow of information from a low speed BIST 100, through the command multiplier (CMX) 110 to a memory system 120. During self-test, the BIST generates at low speed (f_{BIST}) the seed set of CAD information, such as, commands, addresses, and data, which are inputs to the CMX. This seed set of CAD information is used to generate n sets of CAD information which are output to the memory at a high speed (f_{MEM}), where

[0027] $f_{MEM} = f_{BIST} \times n$

[0028] where: f_{MEM} is the operating frequency of the memory,

[0029] f_{BIST} is the operating frequency of the BIST,

[0030] n is the frequency multiplier.

[0031] This invention can be realized by several different configurations or embodiments shown in Figures 2,3,4,5, and 6. Generally, configurations shown in Figures 2-5 generate

the CAD sets in a bit-wise fashion by combining the seed CAD set with the contents of a directive register 130 through logic unit LU to a multiplexer 150. The configuration shown in Figure 6 generates the "n" CAD sets based on offsets which are combined with the seed set in an ALU 140.

[0032] Note that the contents of the CMX registers 140 in the command multiplier 110, which could be directives or offset registers, switch at frequencies less than f_{BIST} and typically will be static for an entire set of patterns generated by the BIST. As indicated in Figure 1, these registers can be loaded via a dedicated, low-speed bus (REG_BUS) or by any other low speed means such as, for example, by an external tester.

[0033] Bit-wise CAD Generation

[0034] For bit-wise generation n clocks of frequency f_{BIST} are provided as inputs to the command multiplier shown in Figures 2-5, each of which is 360/n degrees out of phase from another. That is, if the i'th input clock is known as $CLK_{in,i}$, then $CLK_{in,i+1}$ is advanced by 360 / n degrees from $CLK_{in,i}$ for all i =2 to n. The duty cycle of the input clocks should be 1 / (2 x n). Such clocks may be generated from an external tester, or from an on-chip DLL circuit.

[0035]   The command multiplier 110 output clock is derived from the logical-or'ing function of all $CLK_{in,i}$ signals. Hence, the active edge of each $CLK_{in,i}$ signal identifies the start of the memory cycle. Thus, there will be "n" memory cycles associated with each BIST cycle.

[0036]   With each CAD signal is associated from 1 to "n" directives that are stored in the directive registers. Each directive may consist of one or more bits. Directive registers switch at frequencies less than f_{BIST} and typically will be static for an entire set of patterns generated by the BIST. The directives for a given CAD signal will be used to determine the output CAD set during each of the n embedded memory cycles that occur during a particular BIST cycle based on the output of the BIST and the state of the directive registers. Hence the output of the command multiplier is:

[0037]   CAD_out(i, m) = function(CAD_in(i), Directives(i), m)

[0038]   where

[0039]   CAD_out(i, m) is the i'th output CAD bit during the m'th of n embedded memory cycles associated with a BIST cycle,

[0040]   function is a logical function,

[0041]   CAD_in(i) is the i'th input CAD bit which is supplied by the BIST and may change each BIST cycle,

[0042] Directives(i) are input directives associated with the i'th CAD bit. They are constant for one to many BIST cycles,

[0043] m describes one of the n embedded memory cycles that make up one BIST cycle.

[0044] For example, there may be n directives, with each directive consisting of one bit. If the m'th directive bit is set in the m generator 160 that may be input to the multiplexer 150 or register 130, this could indicate that the CAD output is the inverse of the CAD input. If the directive bit is not set, then the CAD output is to equal the CAD input. Logically this is described as:

[0045] $CAD\_out(i, m) = CAD\_in(i) \text{ XOR } Directive(i, m)$

[0046] where

[0047] Directive(i, m) describes the m'th directive bit for the i'th CAD signal. Such a scheme can be implemented in a circuit as shown in Fig. 2.

[0048] An alternate embodiment is shown in Fig. 3, where the embedded memory cycle is an input to the logical function. Still other embodiments are shown in Fig. 4 and 5.

[0049] In general, each Directive register(i, n) may be any number of bits. More bits may be used to enable more complicated or more flexible logical functions. Examples of some

functions that may be performed using the directives are given below.

[0050] Example 1:

[0051] As shown in the embodiment in Fig. 2 where each Directive(i, m) is a single bit.

[0052] Logical function = Directive(i, m) XOR CAD_in(i)

[0053] Example 2:

[0054] As shown in the embodiment in Fig. 2 where each Directive is two bits,

[0055] Directive_A(i, m) and Directive_B(i, m).

[0056] Logical function = ((Directive_A(i, m) AND Directive_B(i, m)) OR

[0057] (Directive_A(i, m) XOR CAD_in(i))) AND

[0058] (Directive_A(i, m) OR Directive_B(i, m))

[0059] Example 3:

[0060] As shown in the embodiment of Figs. 3 or 4 where "n" is four, and the m in-put to the command multiplier is en-coded as a 2-bit word made of bits m_1 and m_2.

[0061] Also let each directive be one bit.

[0062] Logical Function = CAD_in(i) XOR (Directive(i, m) XOR m_1)

XOR

[0063] (Directive(i, m) XOR m_2)

[0064] Example 4:

[0065] The behavior of Example 3 may be realized using the embodiment of Fig. 5. However, in this case, by utilizing the 2-bit encoding of the m variable, the number of directives can be reduced to two, with each directive consisting of 1-bit. In general, it is often possible to have fewer than m directives in this configuration.

[0066] Logical function = CAD_in(i) XOR (Directive(i, 1) XOR m_1) XOR

[0067] (Directive(i, 2) XOR m_2)

[0068] In addition to the circuitry described above, delays and pipeline stages must be added to assure that the CAD_out signals are synchronized with the output clock.

[0069] CAD Generation Based on Offsets

[0070] A memory which supports concurrent refresh (CCR) requires two bank addresses for each high-speed cycle, one for the command and one for the refresh. This means that, at the same time a READ, WRITE, or no-operation (NOP) command is being performed to bank address BA, a refresh is being performed to bank address RBA, which

was generated using the same seed. Alternatively, separate seeds could be used to generate the n bank addresses for the command and CCR. A non-bit-wise addressing sequence may be accomplished by an offset addressing approach. As indicated above, the CMX 110 in Fig.1 may include an offset address register and an ALU to generate the subsequent high-speed addresses. The detailed operation is described below.

[0071] Prior to the actual self-test, the BIST loads the n sets of CMX registers 130 via the REG_BUS. The register bits in the CMX contain the offset used to generate the subsequent high-speed addresses. When the BIST engine generates the seed CAD, the seed CAD information is captured by the CMX in synch with the low-speed clock (CLK$_{BIST}$). The subsequent addresses for each command and refresh are then calculated by the adder circuit using the seed address and the corresponding address offset. The addresses are then launched to the memory in synch with the high-speed clock (CLK$_{MEM}$).

[0072] The offset addresses may or may not be constant for each clock cycle. For example, by utilizing one seed bank address, BA, and two sets of n bank address offsets, OFFSET1 to OFFSETn for the command and OFFSETR1 to OFF-

SETRn for refresh, the CMX can generate n high-speed bank addresses and n high-speed CCR addresses:

[0073] Command Addresses CCR Addresses

[0074] BA+OFFSET1 BA + OFFSETR1

[0075] BA+OFFSET2 BA + OFFSETR2

[0076] ...

[0077] BA + OFFSETn BA + OFFSETRn

[0078] Because OFFSETi and OFFSETRi may be any integer, any addressing sequence can be generated. As discussed in the background, the most typical example is the address generation of b, b+1, b+2, ... b+n, where b is any integer, which enables a worst-case noise pattern in a multi-bank operation. For example, a command operation and a CCR operation can be issued to two adjacent banks, b and b+1,simultaneously.

[0079] The number of OFFSET registers can be increased to allow the testing of multi-port memories.

[0080] Since the address space of a memory is finite, the result of the addition of a bank address seed and an offset will typically be truncated such that any binary carry that would produce an address that is outside of the address space of

the memory is ignored.

[0081] Alternatively, the subsequent address generation may be determined by not only the seed and the contents of the CMX register's offset address, but also previous CAD seed inputs. For example, in order to perform a refresh on every other high-speed cycle, a counter could keep track of the number of refreshes so that the first eight refreshes to a 16 bank memory would be to the even banks, the next eight would be to odd banks, then even banks, then odd banks, and so on.

[0082] Figure 6 illustrates another embodiment of CAD generation in which:

[0083] ·n = 4;

[0084] ·the CMX generates the word address WA, the bank address BA, and the refresh bank address RBA by adding an offset (contained in the CMX registers) to the seed addresses provided by the BIST;

[0085] ·only two data bits (DATA) are generated, one for all even data inputs of the memory, and one for all odd data inputs; these are generated based on op codes loaded into the CMX;

[0086] ·the high-speed command information (CMD) is generated based on op codes loaded into the CMX.

[0087] The command and data op codes are provided in Figure 7, Tables 1 and 2, respectively. While the invention has been described in terms of preferred embodiments, those skilled in the art will recognize that the invention can be practiced with modifications within the spirit and scope of the amended claims.